Matthew Ridder,
Doug Etts,
Joe Fackler

Design Review 1: Team NOR

Design Review 1 Progress

In this first design sprint, we successfully completed the AND, OR, PASS A, and 8:1 MUX components for the ALU. Because the AND, OR, and PASS A components all had to be 16 bits wide, we had to use some smaller components from the ECE3663 Library including a 2-bit NAND gate, a 2-bit NOR gate, a 2-bit AND gate, a 2-bit OR gate, an inverter, and a transmission gate. Using these components, we created the 16-bit AND gate by putting 16 2-bit AND gates in parallel. Next, we created the 16-bit OR gate by putting 16 2-bit OR gates in parallel. Lastly, we created the 16-bit PASS A component by putting 16 transmission gates in parallel. After completing these 16-bit components, we tested to make sure that they had the correct functionality by simulating one of the bit locations for each component in Ocean. For each component we set up a test bench that had the inputs driven by two series inverters while the output drove an inverter with four times the minimum W/L. We got successful results for each component.

After building the 16-bit components we created the 8:1 MUX. This gate was a bit trickier since it needed inputs and select lines that determine which input gets passed through to the output. To build this gate we used some components from the ECE3663 Library such as 3 and 4 input NAND gates. We also designed our own 2:1 MUX and 4:1 MUX using these components because we found that the Muxes provided in the library were undesirable. With these Muxes we were able to complete our 8:1 MUX, and we simulated it for each select bit position using a similar test bench setup as used for our previous components. We got successful results for this component as well.

All of our simulation results and netlists are provided later in this Design Review. We chose to provide netlists instead of the schematics because we believe that netlists are more compact, more readable, and give a better description of the internal structure of the gates.

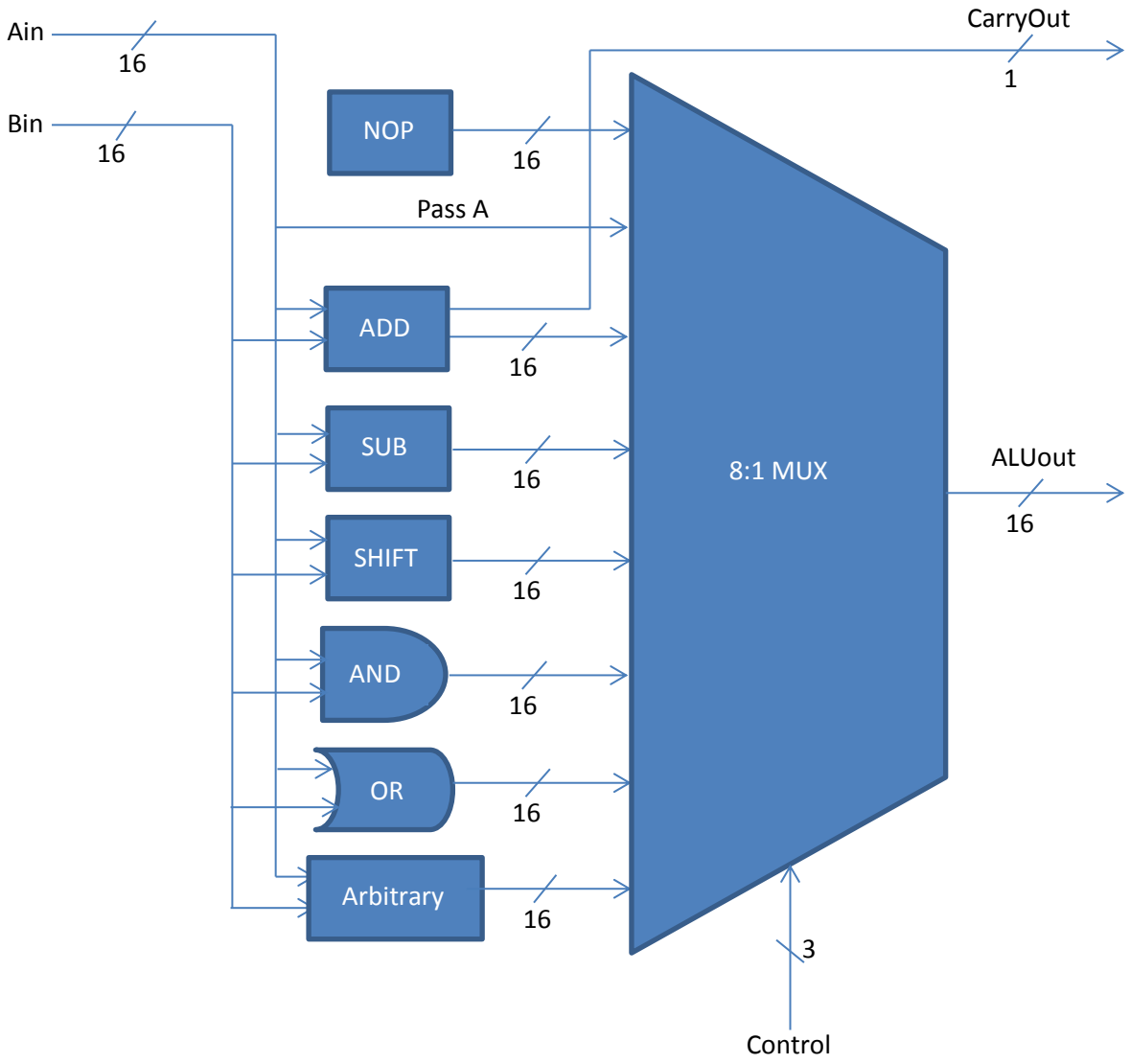
Future Tasks

Prior to the next design review we should have the ADD component, the SUB component, the SHIFT component, the ALU in/out connectivity, and the resisters working. We should also decide on a metric to optimize and design and modify our components accordingly. Lastly, we should decide on an arbitrary function to implement. As of now, some of our ideas for the arbitrary function include a 16-bit comparator that would pass the larger 16-bit value through to the MUX, or a Barrel Shifter that would shift and rotate the bits in multiple directions. We think that both of these functions would give our team a competitive advantage over other teams since they are both invaluable in the design of computer processors.

After the next design review, we should add finishing touches as needed to our project. We should also have a 4 page conference-style report written about our project, and must be prepared to present to the class.

Matthew Ridder,
Doug Etts,
Joe Fackler

Here is the Block Diagram for our ALU:



Matthew Ridder,
Doug Etts,
Joe Fackler

AND Gate Netlist:

```
// Cell name: TNOR_AND
// Function: 16-bit AND
// Inputs: A<15:0> B<15:0>
// Outputs: out<15:0>
subckt TNOR_AND VDD VSS A0 A1 A2 A3 A4 A5 A6 A7 A8 A9 A10 A11 \
       A12 A13 A14 A15 B0 B1 B2 B3 B4 B5 B6 B7 B8 B9 B10 \
       B11 B12 B13 B14 B15 out0 out1 out2 out3 out4 out5 \
       out6 out7 out8 out9 out10 out11 out12 out13 out14 out15
  I0 (VDD VSS A0 B0 out0) TNOR_AND2
  I1 (VDD VSS A1 B1 out1) TNOR_AND2
  I2 (VDD VSS A2 B2 out2) TNOR_AND2
  I3 (VDD VSS A3 B3 out3) TNOR_AND2
  I4 (VDD VSS A4 B4 out4) TNOR_AND2
  I5 (VDD VSS A5 B5 out5) TNOR_AND2
  I6 (VDD VSS A6 B6 out6) TNOR_AND2
  I7 (VDD VSS A7 B7 out7) TNOR_AND2
  I8 (VDD VSS A8 B8 out8) TNOR_AND2
  I9 (VDD VSS A9 B9 out9) TNOR_AND2
  I10 (VDD VSS A10 B10 out10) TNOR_AND2
  I11 (VDD VSS A11 B11 out11) TNOR_AND2
  I12 (VDD VSS A12 B12 out12) TNOR_AND2
  I13 (VDD VSS A13 B13 out13) TNOR_AND2
  I14 (VDD VSS A14 B14 out14) TNOR_AND2
  I15 (VDD VSS A15 B15 out15) TNOR_AND2
ends TNOR_AND
// End of subcircuit definition
```

OR Gate Netlist:

```
// Function: 16-bit OR
// Inputs: A<15:0> B<15:0>
// Outputs: out<15:0>
subckt TNOR_OR VDD VSS A0 A1 A2 A3 A4 A5 A6 A7 A8 A9 A10 A11 \
       A12 A13 A14 A15 B0 B1 B2 B3 B4 B5 B6 B7 B8 B9 B10 \
       B11 B12 B13 B14 B15 out0 out1 out2 out3 out4 out5 \
       out6 out7 out8 out9 out10 out11 out12 out13 out14 out15
  I0 (VDD VSS A0 B0 out0) TNOR_OR2
  I1 (VDD VSS A1 B1 out1) TNOR_OR2
  I2 (VDD VSS A2 B2 out2) TNOR_OR2
  I3 (VDD VSS A3 B3 out3) TNOR_OR2
  I4 (VDD VSS A4 B4 out4) TNOR_OR2
  I5 (VDD VSS A5 B5 out5) TNOR_OR2
  I6 (VDD VSS A6 B6 out6) TNOR_OR2
  I7 (VDD VSS A7 B7 out7) TNOR_OR2
  I8 (VDD VSS A8 B8 out8) TNOR_OR2
  I9 (VDD VSS A9 B9 out9) TNOR_OR2
  I10 (VDD VSS A10 B10 out10) TNOR_OR2
  I11 (VDD VSS A11 B11 out11) TNOR_OR2
  I12 (VDD VSS A12 B12 out12) TNOR_OR2
  I13 (VDD VSS A13 B13 out13) TNOR_OR2
```

```
  I14 (VDD VSS A14 B14 out14) TNOR_OR2
  I15 (VDD VSS A15 B15 out15) TNOR_OR2
ends TNOR_OR
// End of subcircuit definition
```

PASS A Netlist:

```
// Cell name: TNOR_PASSA
// Function: 16-bit PASS
// Inputs: A<15:0> B<15:0>
// Outputs: out<15:0>
subckt TNOR_PASSA VDD VSS A0 A1 A2 A3 A4 A5 A6 A7 A8 A9 A10 A11 \
       A12 A13 A14 A15 control out0 out1 out2 out3 out4 out5 \
       out6 out7 out8 out9 out10 out11 out12 out13 out14 out15
  I0 (VDD VSS A0 control out0) TNOR_PASSA1
  I1 (VDD VSS A1 control out1) TNOR_PASSA1
  I2 (VDD VSS A2 control out2) TNOR_PASSA1
  I3 (VDD VSS A3 control out3) TNOR_PASSA1
  I4 (VDD VSS A4 control out4) TNOR_PASSA1
  I5 (VDD VSS A5 control out5) TNOR_PASSA1
  I6 (VDD VSS A6 control out6) TNOR_PASSA1
  I7 (VDD VSS A7 control out7) TNOR_PASSA1
  I8 (VDD VSS A8 control out8) TNOR_PASSA1
  I9 (VDD VSS A9 control out9) TNOR_PASSA1
  I10 (VDD VSS A10 control out10) TNOR_PASSA1
  I11 (VDD VSS A11 control out11) TNOR_PASSA1
  I12 (VDD VSS A12 control out12) TNOR_PASSA1
  I13 (VDD VSS A13 control out13) TNOR_PASSA1
  I14 (VDD VSS A14 control out14) TNOR_PASSA1
  I15 (VDD VSS A15 control out15) TNOR_PASSA1
ends TNOR_PASSA
// End of subcircuit definition
```

8:1 MUX Netlist:

*For this Netlist we included the 2:1 MUX and 4:1 MUX subcircuit definitions because we used our own designs instead of the ones in the ECE3663 Library.

```
// Cell Name: TNOR_MUX2to1
// Function: 2:1 Multiplexer
// Inputs: A B S0
// Output: out
// Relies on TNOR_Inverter and TNOR_NAND2
subckt TNOR_MUX2to1 VDD VSS A B S out
I0 (VDD VSS S net0) TNOR_Inverter
I1 (VDD VSS A net0 net1) TNOR_NAND2
I2 (VDD VSS B S net2) TNOR_NAND2
I3 (VDD VSS net1 net2 out) TNOR_NAND2
ends TNOR_MUX2to1
// End of subckt def
```

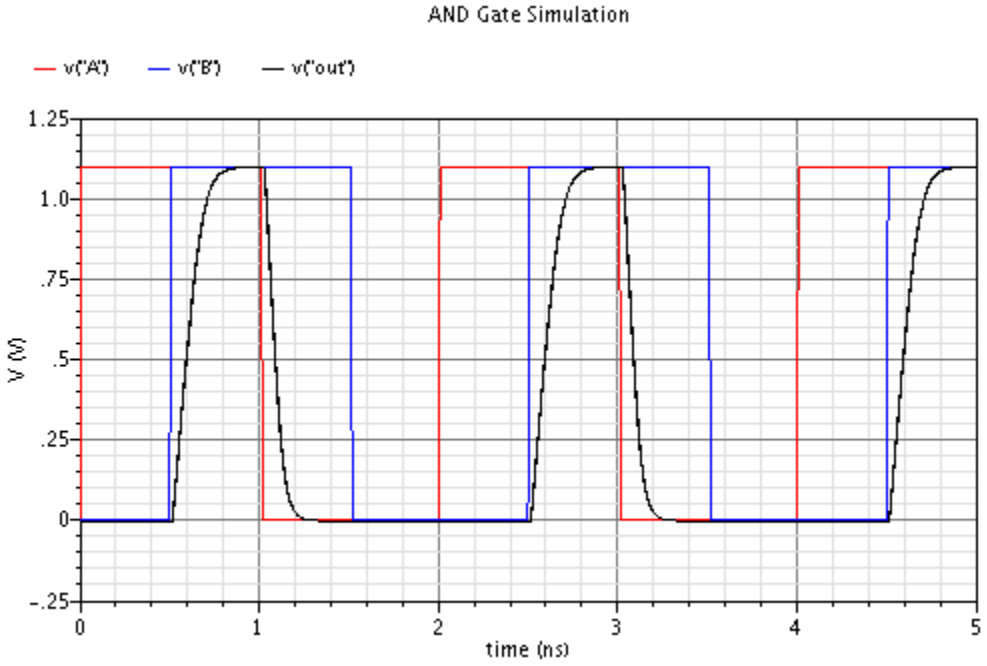
Matthew Ridder,
Doug Etts,
Joe Fackler

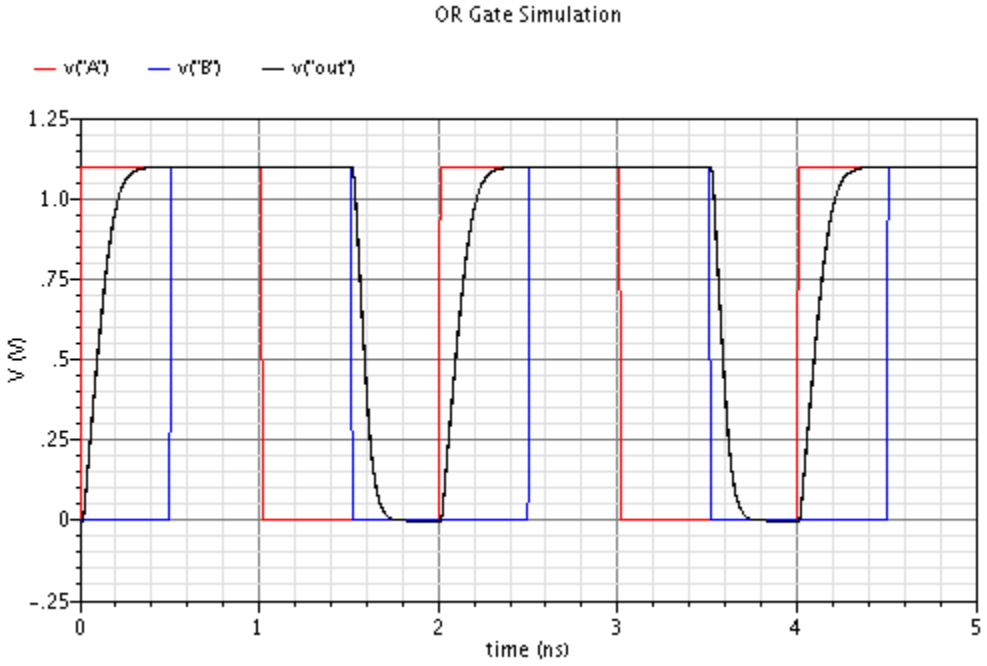
```
// Cell Name: TNOR_MUX4to1
// Function: 4:1 Multiplexer
// Inputs: A B C D S0 S1
// Output: out
// Relies on TNOR_Inverter, TNOR_NAND3, and TNOR_NAND4
subckt TNOR_MUX4to1 VDD VSS A B C D S0 S1 out
I0 (VDD VSS S0 net0) TNOR_Inverter
I1 (VDD VSS S1 net1) TNOR_Inverter
I2 (VDD VSS A net0 net1 net2) TNOR_NAND3
I3 (VDD VSS B S0 net1 net3) TNOR_NAND3
I4 (VDD VSS C net0 S1 net4) TNOR_NAND3
I5 (VDD VSS D S0 S1 net5) TNOR_NAND3
I6 (VDD VSS net2 net3 net4 net5 out) TNOR_NAND4
ends TNOR_MUX4to1
// Ends subckt def

// Cell Name: TNOR_MUX8to1
// 8:1 Multiplexer
// Inputs: A B C D E F G H S0 S1 S2
// Output: out
// Relies on TNOR_MUX2to1 and TNOR_MUX4to1
subckt TNOR_MUX8to1 VDD VSS A B C D E F G H S0 S1 S2 out
I0 (VDD VSS A B C D S0 S1 net0) TNOR_MUX4to1
I1 (VDD VSS E F G H S0 S1 net1) TNOR_MUX4to1
I2 (VDD VSS net0 net1 S2 out) TNOR_MUX2to1
ends TNOR_MUX8to1
// Ends subckt def
```

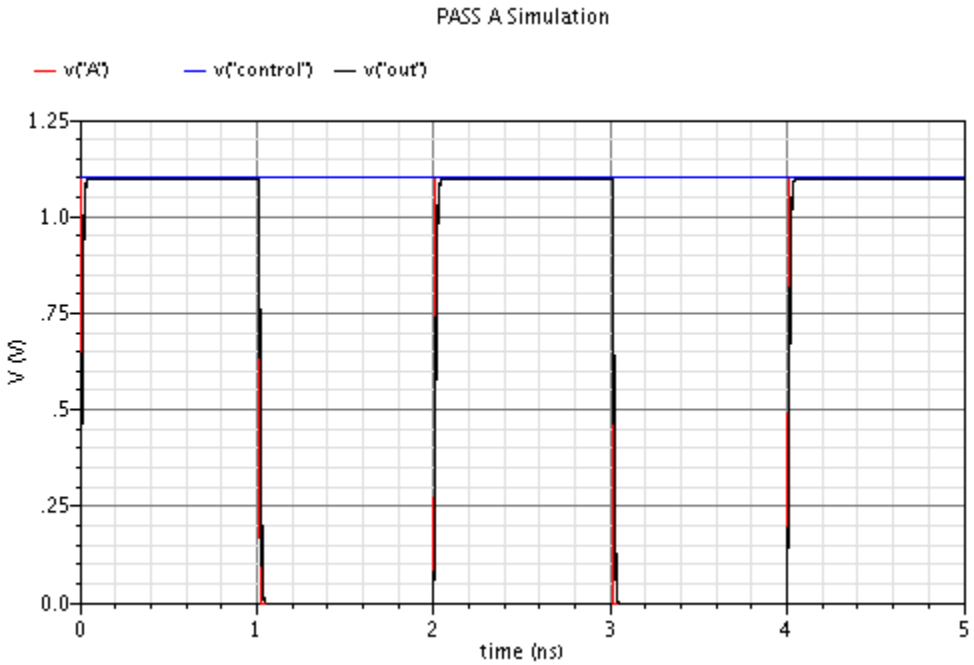
Matthew Ridder,
Doug Etts,
Joe Fackler

Simulation Results

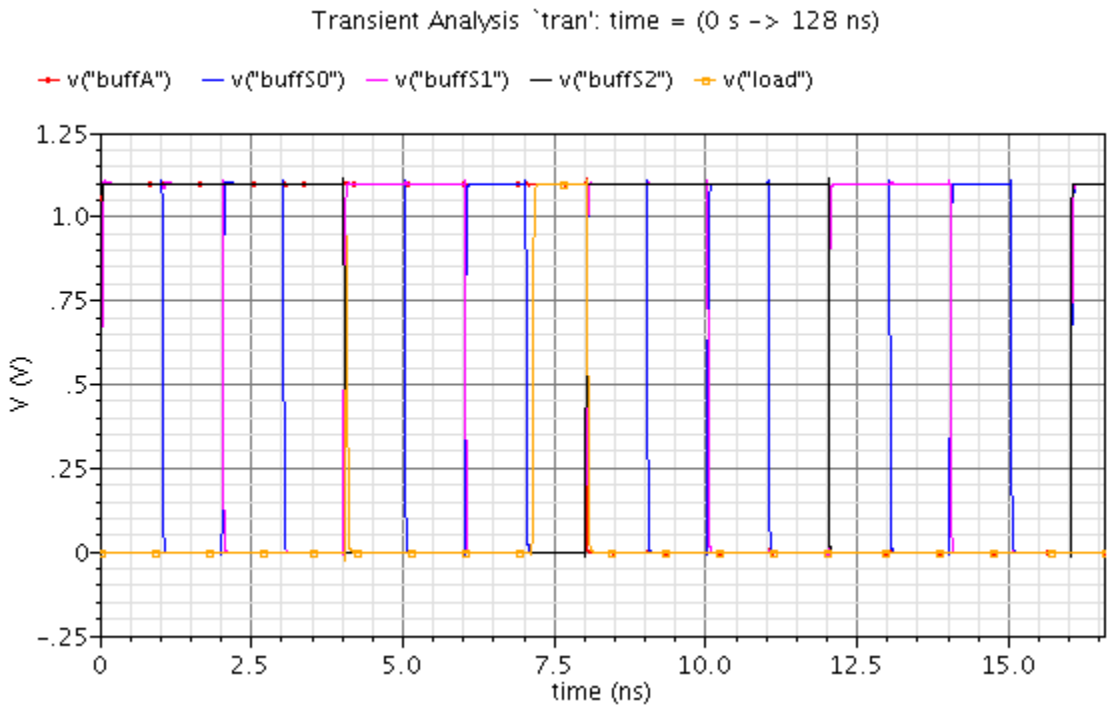




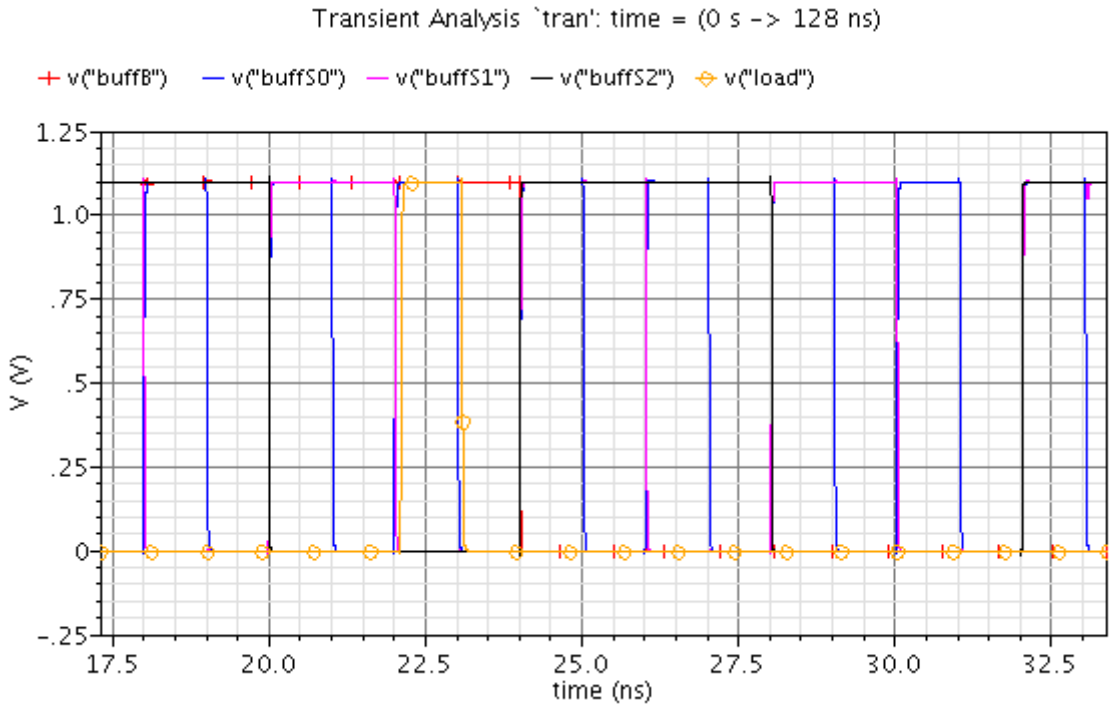
Matthew Ridder,
Doug Etts,
Joe Fackler

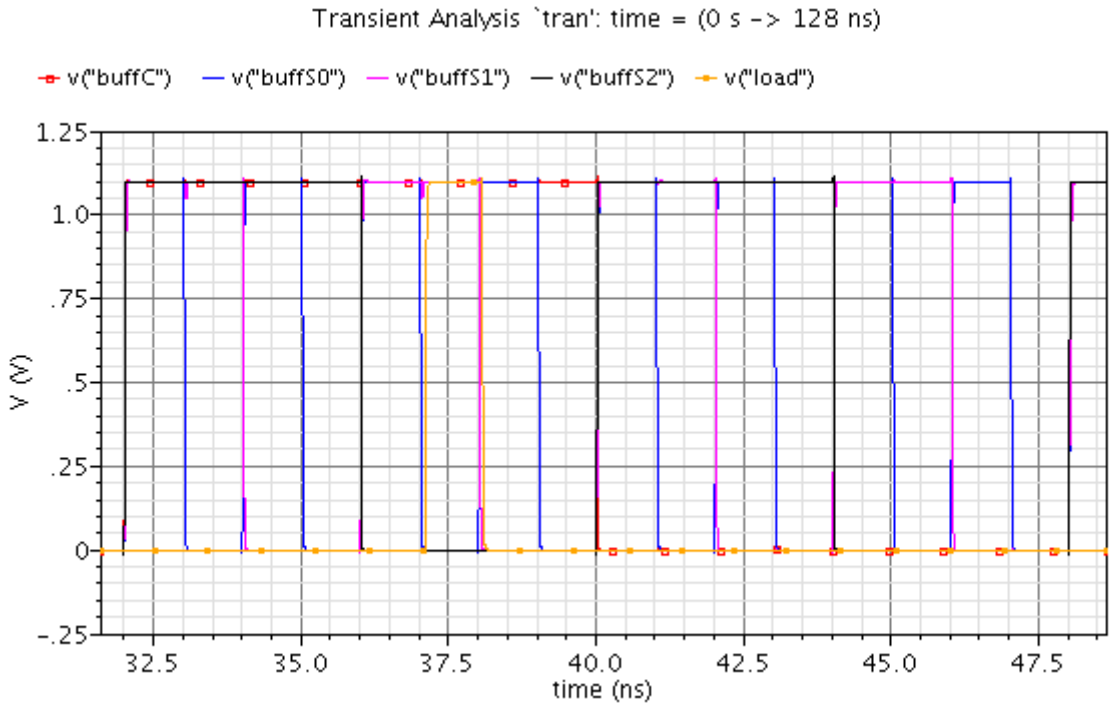


8:1 MUX Plots for inputs A to H:

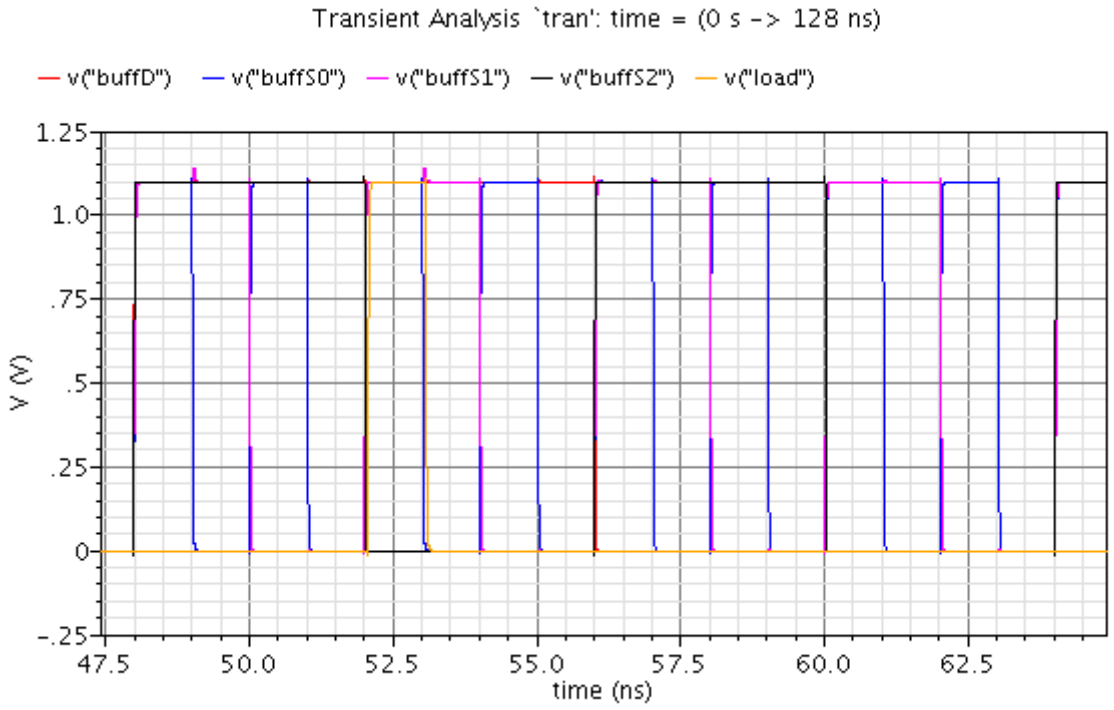


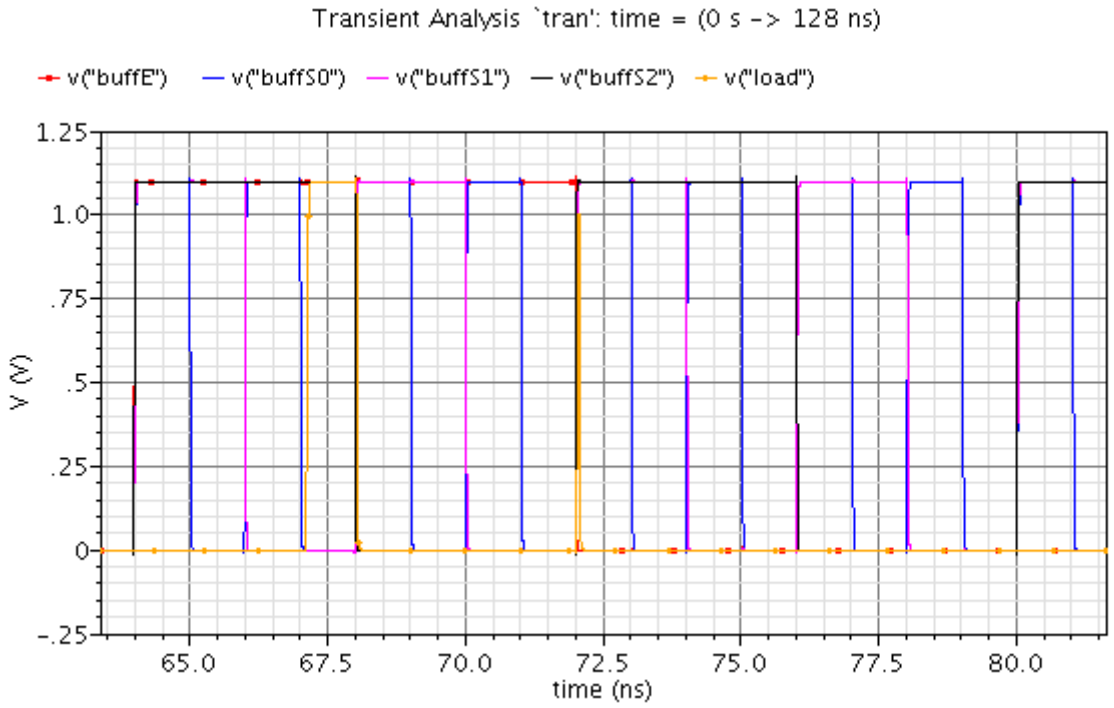
Matthew Ridder,
Doug Etts,
Joe Fackler



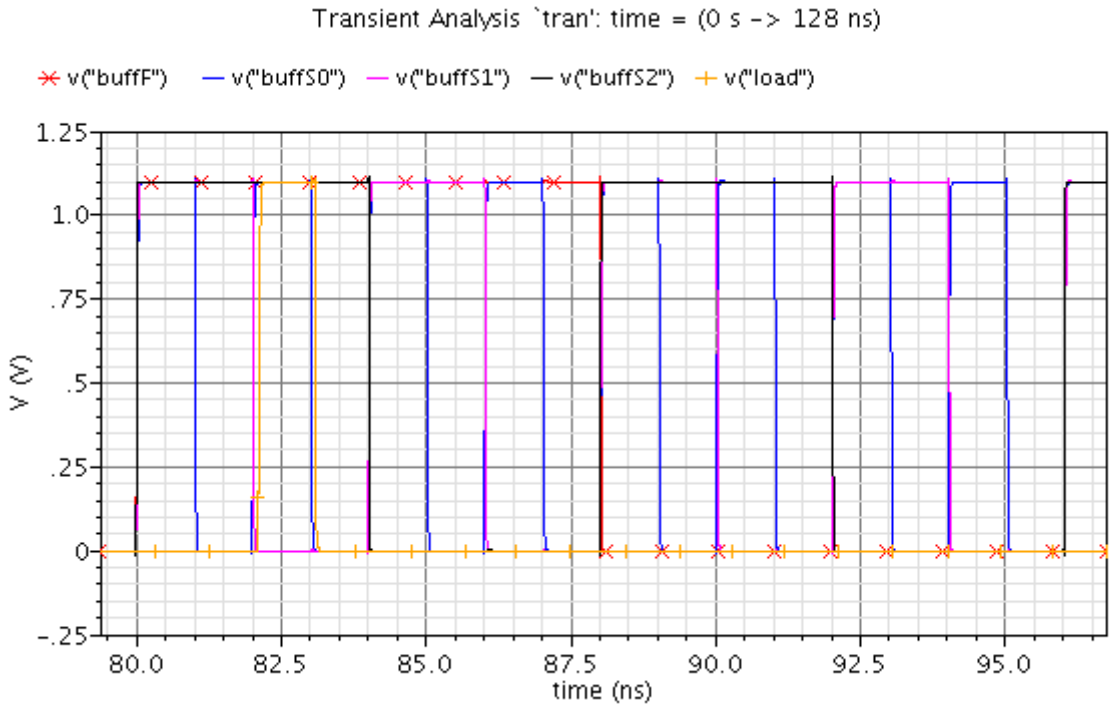


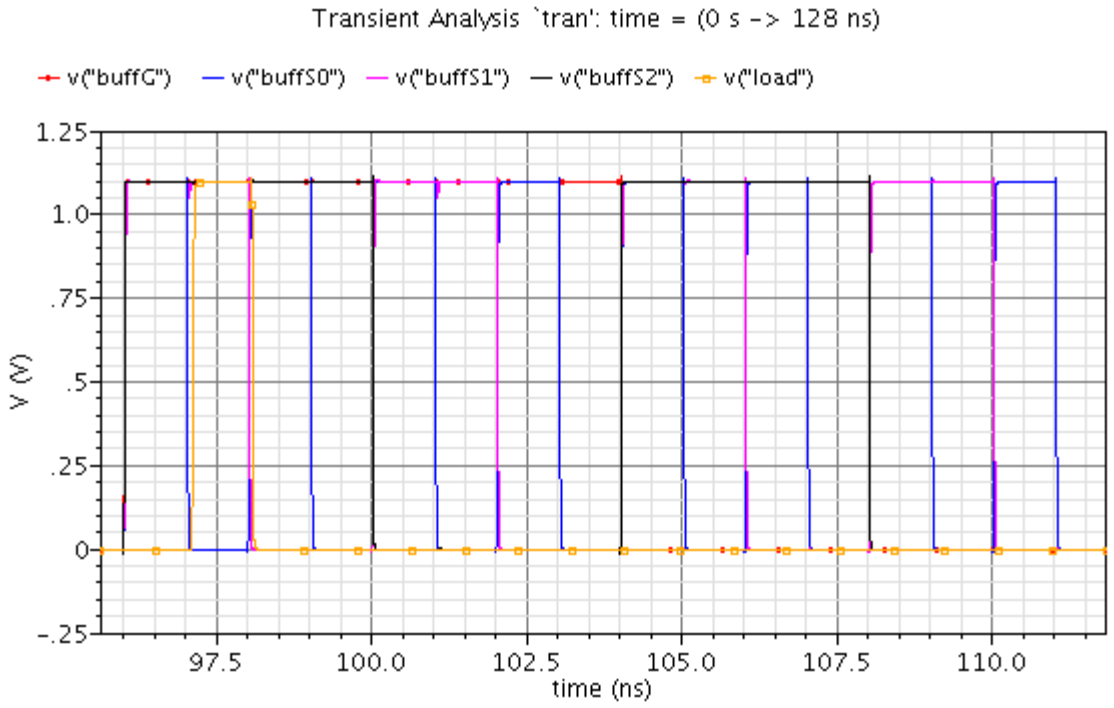
Matthew Ridder,
Doug Etts,
Joe Fackler





Matthew Ridder,
Doug Etts,
Joe Fackler





Matthew Ridder,
Doug Etts,
Joe Fackler